**DAY 06**

**24.06.2023**

**Dataset**

Architeratural Design dataset (multi-class dataset) have been shared with ten s namely altar, apse, bell\_tower, column, dome(inner), dome(outer), flying buttress, gargoyle, stained\_glass, vault.

Train set : train

Test set : test

*Google Drive*

<https://drive.google.com/drive/folders/1CowMN9zfyOzgF46UsqQ-j7xs90r8NjCC?usp=drive_link>

**Convolutional Neural Network**

A convolutional neural network (CNN), is a network architecture for deep learning which learns directly from the data.

*CNN Architecture*

* VGG
* Xception
* ResNet
* InceptionV3
* InceptionResNet
* MobileNet
* DenseNet
* NasNet
* EfficientNet
* ConvNEXT

*Reference*

<https://towardsdatascience.com/top-10-pre-trained-models-for-image-embedding-every-data-scientist-should-know-88da0ef541cd>

<https://www.tensorflow.org/api_docs/python/tf/keras/applications/densenet/DenseNet169>

**Modules**

[**activations**](https://www.tensorflow.org/api_docs/python/tf/keras/activations)**module:** Built-in activation functions.

[**applications**](https://www.tensorflow.org/api_docs/python/tf/keras/applications)**module:** Keras Applications are premade architectures with pre-trained weights.

[**backend**](https://www.tensorflow.org/api_docs/python/tf/keras/backend)**module**: Keras backend API.

[**callbacks**](https://www.tensorflow.org/api_docs/python/tf/keras/callbacks)**module**: Callbacks: utilities called at certain points during model training.

[**constraints**](https://www.tensorflow.org/api_docs/python/tf/keras/constraints)**module**: Constraints: functions that impose constraints on weight values.

[**datasets**](https://www.tensorflow.org/api_docs/python/tf/keras/datasets)**module:** Small NumPy datasets for debugging/testing.

[**dtensor**](https://www.tensorflow.org/api_docs/python/tf/keras/dtensor)**module**: Keras' DTensor library.

[**estimator**](https://www.tensorflow.org/api_docs/python/tf/keras/estimator)**module**: Keras estimator API.

[**experimental**](https://www.tensorflow.org/api_docs/python/tf/keras/experimental)**module:** Public API for tf.keras.experimental namespace.

[**export**](https://www.tensorflow.org/api_docs/python/tf/keras/export)**module:** Public API for tf.keras.export namespace.

[**initializers**](https://www.tensorflow.org/api_docs/python/tf/keras/initializers)**module:** Keras initializer serialization / deserialization.

[**layers**](https://www.tensorflow.org/api_docs/python/tf/keras/layers)**module:** Keras layers API.

[**losses**](https://www.tensorflow.org/api_docs/python/tf/keras/losses)**module:** Built-in loss functions.

[**metrics**](https://www.tensorflow.org/api_docs/python/tf/keras/metrics)**module:** All Keras metrics.

[**mixed\_precision**](https://www.tensorflow.org/api_docs/python/tf/keras/mixed_precision)**module:** Keras mixed precision API.

[**models**](https://www.tensorflow.org/api_docs/python/tf/keras/models)**module:** Keras models API.

[**optimizers**](https://www.tensorflow.org/api_docs/python/tf/keras/optimizers)**module:** Built-in optimizer classes.

[**preprocessing**](https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing)**module:** Utilities to preprocess data before training.

[**regularizers**](https://www.tensorflow.org/api_docs/python/tf/keras/regularizers)**module:** Built-in regularizers.

[**saving**](https://www.tensorflow.org/api_docs/python/tf/keras/saving)**module:** Public API for tf.keras.saving namespace.

[**utils**](https://www.tensorflow.org/api_docs/python/tf/keras/utils)**module:** Public Keras utilities.

## Functions

[deserialize(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/deserialize): Returns activation function given a string identifier.

[elu(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/elu): Exponential Linear Unit.

[exponential(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/exponential): Exponential activation function.

[gelu(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/gelu): Applies the Gaussian error linear unit (GELU) activation function.

[get(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/get): Returns function.

[hard\_sigmoid(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/hard_sigmoid): Hard sigmoid activation function.

[linear(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/linear): Linear activation function (pass-through).

[relu(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/relu): Applies the rectified linear unit activation function.

[selu(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/selu): Scaled Exponential Linear Unit (SELU).

[serialize(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/serialize): Returns the string identifier of an activation function.

[sigmoid(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/sigmoid): Sigmoid activation function, sigmoid(x) = 1 / (1 + exp(-x)).

[softmax(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/softmax): Softmax converts a vector of values to a probability distribution.

[softplus(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/softplus): Softplus activation function, softplus(x) = log(exp(x) + 1).

[softsign(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/softsign): Softsign activation function, softsign(x) = x / (abs(x) + 1).

[swish(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/swish): Swish activation function, swish(x) = x \* sigmoid(x).

[tanh(...)](https://www.tensorflow.org/api_docs/python/tf/keras/activations/tanh): Hyperbolic tangent activation function.

## Classes

[class Constraint](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/Constraint): Base class for weight constraints.

[class MaxNorm](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/MaxNorm): MaxNorm weight constraint.

[class MinMaxNorm](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/MinMaxNorm): MinMaxNorm weight constraint.

[class NonNeg](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/NonNeg): Constrains the weights to be non-negative.

[class RadialConstraint](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/RadialConstraint): Constrains Conv2D kernel weights to be the same for each radius.

[class UnitNorm](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/UnitNorm): Constrains the weights incident to each hidden unit to have unit norm.

[class max\_norm](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/MaxNorm): MaxNorm weight constraint.

[class min\_max\_norm](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/MinMaxNorm): MinMaxNorm weight constraint.

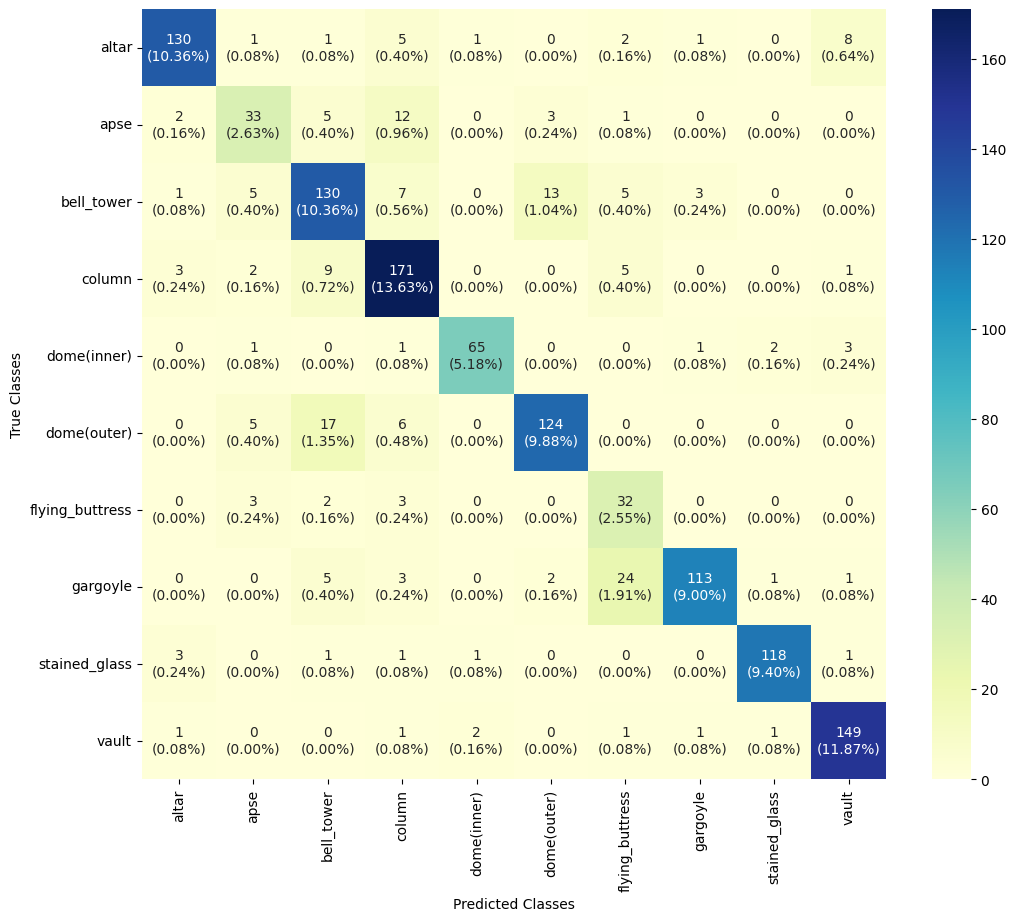
[class non\_neg](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/NonNeg): Constrains the weights to be non-negative.

[class radial\_constraint](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/RadialConstraint): Constrains Conv2D kernel weights to be the same for each radius.

[class unit\_norm](https://www.tensorflow.org/api_docs/python/tf/keras/constraints/UnitNorm): Constrains the weights incident to each hidden unit to have unit norm.

**VGG**

*Confusion Matrix*



**Epoch :** 100

**Batch size :** 64

**LR :** 0.001

**Accuracy :** 0.811952

**Precision :** 0.770157

**Recall :** 0.805295

**F1 score** : 0.755556